ECGR_5103_Homework6

April 28, 2023

1 ECGR Homework 6

Patrick Flynn | 801055057

Set everything up:

[ ]:
```
!pip install torch torchvision
!pip install d2l==1.0.0a1.post0
!pip install matplotlib_inline
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Requirement already satisfied: torch in /usr/local/lib/python3.9/dist-packages
(2.0.0+cu118)
Requirement already satisfied: torchvision in /usr/local/lib/python3.9/dist-
packages (0.15.1+cu118)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.9/dist-packages
(from torch) (3.1.2)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.9/dist-packages (from torch) (4.5.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.9/dist-packages
(from torch) (1.11.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.9/dist-
packages (from torch) (3.12.0)
Requirement already satisfied: networkx in /usr/local/lib/python3.9/dist-
packages (from torch) (3.1)
Requirement already satisfied: triton==2.0.0 in /usr/local/lib/python3.9/dist-
packages (from torch) (2.0.0)
Requirement already satisfied: cmake in /usr/local/lib/python3.9/dist-packages
(from triton==2.0.0->torch) (3.25.2)
Requirement already satisfied: lit in /usr/local/lib/python3.9/dist-packages
(from triton==2.0.0->torch) (16.0.2)
Requirement already satisfied: requests in /usr/local/lib/python3.9/dist-
packages (from torchvision) (2.27.1)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in
/usr/local/lib/python3.9/dist-packages (from torchvision) (8.4.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.9/dist-packages
(from torchvision) (1.22.4)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.9/dist-
```

```
packages (from jinja2->torch) (2.1.2)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.9/dist-packages (from requests->torchvision) (2022.12.7)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/usr/local/lib/python3.9/dist-packages (from requests->torchvision) (2.0.12)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-
packages (from requests->torchvision) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/usr/local/lib/python3.9/dist-packages (from requests->torchvision) (1.26.15)
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.9/dist-
packages (from sympy->torch) (1.3.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Collecting d2l==1.0.0a1.post0
  Downloading d2l-1.0.0a1.post0-py3-none-any.whl (93 kB)
                             93.0/93.0 kB
11.9 MB/s eta 0:00:00
Requirement already satisfied: matplotlib in
/usr/local/lib/python3.9/dist-packages (from d2l==1.0.0a1.post0) (3.7.1)
Requirement already satisfied: pandas in /usr/local/lib/python3.9/dist-packages
(from d2l==1.0.0a1.post0) (1.5.3)
Requirement already satisfied: matplotlib-inline in
/usr/local/lib/python3.9/dist-packages (from d2l==1.0.0a1.post0) (0.1.6)
Collecting jupyter
  Downloading jupyter-1.0.0-py2.py3-none-any.whl (2.7 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.9/dist-
packages (from d2l==1.0.0a1.post0) (2.27.1)
Requirement already satisfied: gym in /usr/local/lib/python3.9/dist-packages
(from d2l==1.0.0a1.post0) (0.25.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.9/dist-packages
(from d2l==1.0.0a1.post0) (1.22.4)
Requirement already satisfied: gym-notices>=0.0.4 in
/usr/local/lib/python3.9/dist-packages (from gym->d2l==1.0.0a1.post0) (0.0.8)
Requirement already satisfied: importlib-metadata>=4.8.0 in
/usr/local/lib/python3.9/dist-packages (from gym->d2l==1.0.0a1.post0) (6.6.0)
Requirement already satisfied: cloudpickle>=1.2.0 in
/usr/local/lib/python3.9/dist-packages (from gym->d2l==1.0.0a1.post0) (2.2.1)
Requirement already satisfied: ipykernel in /usr/local/lib/python3.9/dist-
packages (from jupyter->d2l==1.0.0a1.post0) (5.5.6)
Requirement already satisfied: ipywidgets in /usr/local/lib/python3.9/dist-
packages (from jupyter->d2l==1.0.0a1.post0) (7.7.1)
Requirement already satisfied: jupyter-console in /usr/local/lib/python3.9/dist-
packages (from jupyter->d2l==1.0.0a1.post0) (6.1.0)
Requirement already satisfied: nbconvert in /usr/local/lib/python3.9/dist-
packages (from jupyter->d2l==1.0.0a1.post0) (6.5.4)
Collecting qtconsole
  Downloading qtconsole-5.4.2-py3-none-any.whl (121 kB)
                            121.2/121.2 kB
```

```
10.3 MB/s eta 0:00:00
Requirement already satisfied: notebook in /usr/local/lib/python3.9/dist-
packages (from jupyter->d2l==1.0.0a1.post0) (6.4.8)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.9/dist-packages (from matplotlib->d2l==1.0.0a1.post0)
(3.0.9)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.9/dist-
packages (from matplotlib->d2l==1.0.0a1.post0) (8.4.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.9/dist-
packages (from matplotlib->d2l==1.0.0a1.post0) (23.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.9/dist-
packages (from matplotlib->d2l==1.0.0a1.post0) (0.11.0)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.9/dist-packages (from matplotlib->d2l==1.0.0a1.post0)
(1.0.7)
Requirement already satisfied: importlib-resources>=3.2.0 in
/usr/local/lib/python3.9/dist-packages (from matplotlib->d2l==1.0.0a1.post0)
(5.12.0)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.9/dist-packages (from matplotlib->d2l==1.0.0a1.post0)
(4.39.3)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.9/dist-packages (from matplotlib->d2l==1.0.0a1.post0)
(2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.9/dist-packages (from matplotlib->d2l==1.0.0a1.post0)
(1.4.4)
Requirement already satisfied: traitlets in /usr/local/lib/python3.9/dist-
packages (from matplotlib-inline->d2l==1.0.0a1.post0) (5.7.1)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.9/dist-
packages (from pandas->d2l==1.0.0a1.post0) (2022.7.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/usr/local/lib/python3.9/dist-packages (from requests->d2l==1.0.0a1.post0)
(1.26.15)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/usr/local/lib/python3.9/dist-packages (from requests->d2l==1.0.0a1.post0)
(2.0.12)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-
packages (from requests->d2l==1.0.0a1.post0) (3.4)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.9/dist-packages (from requests->d2l==1.0.0a1.post0)
(2022.12.7)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.9/dist-
packages (from importlib-metadata>=4.8.0->gym->d2l==1.0.0a1.post0) (3.15.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-
packages (from python-dateutil>=2.7->matplotlib->d2l==1.0.0a1.post0) (1.16.0)
Requirement already satisfied: jupyter-client in /usr/local/lib/python3.9/dist-
packages (from ipykernel->jupyter->d2l==1.0.0a1.post0) (6.1.12)
```

```
Requirement already satisfied: tornado>=4.2 in /usr/local/lib/python3.9/dist-
packages (from ipykernel->jupyter->d2l==1.0.0a1.post0) (6.2)
Requirement already satisfied: ipython-genutils in
/usr/local/lib/python3.9/dist-packages (from
ipykernel->jupyter->d2l==1.0.0a1.post0) (0.2.0)
Requirement already satisfied: ipython>=5.0.0 in /usr/local/lib/python3.9/dist-
packages (from ipykernel->jupyter->d2l==1.0.0a1.post0) (7.34.0)
Requirement already satisfied: widgetsnbextension~=3.6.0 in
/usr/local/lib/python3.9/dist-packages (from
ipywidgets->jupyter->d2l==1.0.0a1.post0) (3.6.4)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in
/usr/local/lib/python3.9/dist-packages (from
ipywidgets->jupyter->d2l==1.0.0a1.post0) (3.0.7)
Requirement already satisfied: pygments in /usr/local/lib/python3.9/dist-
packages (from jupyter-console->jupyter->d2l==1.0.0a1.post0) (2.14.0)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in
/usr/local/lib/python3.9/dist-packages (from jupyter-
console->jupyter->d2l==1.0.0a1.post0) (3.0.38)
Requirement already satisfied: tinycss2 in /usr/local/lib/python3.9/dist-
packages (from nbconvert->jupyter->d2l==1.0.0a1.post0) (1.2.1)
Requirement already satisfied: lxml in /usr/local/lib/python3.9/dist-packages
(from nbconvert->jupyter->d2l==1.0.0a1.post0) (4.9.2)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.9/dist-
packages (from nbconvert->jupyter->d2l==1.0.0a1.post0) (2.1.2)
Requirement already satisfied: nbclient>=0.5.0 in /usr/local/lib/python3.9/dist-
packages (from nbconvert->jupyter->d2l==1.0.0a1.post0) (0.7.3)
Requirement already satisfied: entrypoints>=0.2.2 in
/usr/local/lib/python3.9/dist-packages (from
nbconvert->jupyter->d2l==1.0.0a1.post0) (0.4)
Requirement already satisfied: mistune<2,>=0.8.1 in
/usr/local/lib/python3.9/dist-packages (from
nbconvert->jupyter->d2l==1.0.0a1.post0) (0.8.4)
Requirement already satisfied: jinja2>=3.0 in /usr/local/lib/python3.9/dist-
packages (from nbconvert->jupyter->d2l==1.0.0a1.post0) (3.1.2)
Requirement already satisfied: nbformat>=5.1 in /usr/local/lib/python3.9/dist-
packages (from nbconvert->jupyter->d2l==1.0.0a1.post0) (5.8.0)
Requirement already satisfied: pandocfilters>=1.4.1 in
/usr/local/lib/python3.9/dist-packages (from
nbconvert->jupyter->d2l==1.0.0a1.post0) (1.5.0)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.9/dist-
packages (from nbconvert->jupyter->d2l==1.0.0a1.post0) (4.11.2)
Requirement already satisfied: defusedxml in /usr/local/lib/python3.9/dist-
packages (from nbconvert->jupyter->d2l==1.0.0a1.post0) (0.7.1)
Requirement already satisfied: bleach in /usr/local/lib/python3.9/dist-packages
(from nbconvert->jupyter->d2l==1.0.0a1.post0) (6.0.0)
Requirement already satisfied: jupyter-core>=4.7 in
/usr/local/lib/python3.9/dist-packages (from
nbconvert->jupyter->d2l==1.0.0a1.post0) (5.3.0)
```

```
Requirement already satisfied: jupyterlab-pygments in
/usr/local/lib/python3.9/dist-packages (from
nbconvert->jupyter->d2l==1.0.0a1.post0) (0.2.2)
Requirement already satisfied: prometheus-client in
/usr/local/lib/python3.9/dist-packages (from
notebook->jupyter->d2l==1.0.0a1.post0) (0.16.0)
Requirement already satisfied: pyzmq>=17 in /usr/local/lib/python3.9/dist-
packages (from notebook->jupyter->d2l==1.0.0a1.post0) (23.2.1)
Requirement already satisfied: nest-asyncio>=1.5 in
/usr/local/lib/python3.9/dist-packages (from
notebook->jupyter->d2l==1.0.0a1.post0) (1.5.6)
Requirement already satisfied: terminado>=0.8.3 in
/usr/local/lib/python3.9/dist-packages (from
notebook->jupyter->d2l==1.0.0a1.post0) (0.17.1)
Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.9/dist-
packages (from notebook->jupyter->d2l==1.0.0a1.post0) (21.3.0)
Requirement already satisfied: Send2Trash>=1.8.0 in
/usr/local/lib/python3.9/dist-packages (from
notebook->jupyter->d2l==1.0.0a1.post0) (1.8.0)
Collecting qtpy>=2.0.1
  Downloading QtPy-2.3.1-py3-none-any.whl (84 kB)
                             84.9/84.9 kB
12.6 MB/s eta 0:00:00
Requirement already satisfied: setuptools>=18.5 in
/usr/local/lib/python3.9/dist-packages (from
ipython>=5.0.0->ipykernel->jupyter->d2l==1.0.0a1.post0) (67.7.2)
Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.9/dist-
packages (from ipython>=5.0.0->ipykernel->jupyter->d2l==1.0.0a1.post0) (4.8.0)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.9/dist-
packages (from ipython>=5.0.0->ipykernel->jupyter->d2l==1.0.0a1.post0) (0.7.5)
Requirement already satisfied: decorator in /usr/local/lib/python3.9/dist-
packages (from ipython>=5.0.0->ipykernel->jupyter->d2l==1.0.0a1.post0) (4.4.2)
Requirement already satisfied: backcall in /usr/local/lib/python3.9/dist-
packages (from ipython>=5.0.0->ipykernel->jupyter->d2l==1.0.0a1.post0) (0.2.0)
Collecting jedi>=0.16
  Downloading jedi-0.18.2-py2.py3-none-any.whl (1.6 MB)
                             1.6/1.6 MB
77.4 MB/s eta 0:00:00
Requirement already satisfied: platformdirs>=2.5 in
/usr/local/lib/python3.9/dist-packages (from jupyter-
core>=4.7->nbconvert->jupyter->d2l==1.0.0a1.post0) (3.2.0)
Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.9/dist-
packages (from nbformat>=5.1->nbconvert->jupyter->d2l==1.0.0a1.post0) (4.3.3)
Requirement already satisfied: fastjsonschema in /usr/local/lib/python3.9/dist-
packages (from nbformat>=5.1->nbconvert->jupyter->d2l==1.0.0a1.post0) (2.16.3)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.9/dist-packages
(from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->jupyter-
console->jupyter->d2l==1.0.0a1.post0) (0.2.6)
```

```
Requirement already satisfied: ptyprocess in /usr/local/lib/python3.9/dist-
packages (from terminado>=0.8.3->notebook->jupyter->d2l==1.0.0a1.post0) (0.7.0)
Requirement already satisfied: argon2-cffi-bindings in
/usr/local/lib/python3.9/dist-packages (from
argon2-cffi->notebook->jupyter->d2l==1.0.0a1.post0) (21.2.0)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.9/dist-
packages (from beautifulsoup4->nbconvert->jupyter->d2l==1.0.0a1.post0) (2.4.1)
Requirement already satisfied: webencodings in /usr/local/lib/python3.9/dist-
packages (from bleach->nbconvert->jupyter->d2l==1.0.0a1.post0) (0.5.1)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in
/usr/local/lib/python3.9/dist-packages (from
jedi>=0.16->ipython>=5.0.0->ipykernel->jupyter->d2l==1.0.0a1.post0) (0.8.3)
Requirement already satisfied: attrs>=17.4.0 in /usr/local/lib/python3.9/dist-
packages (from
jsonschema>=2.6->nbformat>=5.1->nbconvert->jupyter->d2l==1.0.0a1.post0) (23.1.0)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in
/usr/local/lib/python3.9/dist-packages (from
jsonschema>=2.6->nbformat>=5.1->nbconvert->jupyter->d2l==1.0.0a1.post0) (0.19.3)
Requirement already satisfied: cffi>=1.0.1 in /usr/local/lib/python3.9/dist-
packages (from argon2-cffi-
bindings->argon2-cffi->notebook->jupyter->d2l==1.0.0a1.post0) (1.15.1)
Requirement already satisfied: pycparser in /usr/local/lib/python3.9/dist-
packages (from cffi>=1.0.1->argon2-cffi-
bindings->argon2-cffi->notebook->jupyter->d2l==1.0.0a1.post0) (2.21)
Installing collected packages: qtpy, jedi, qtconsole, jupyter, d2l
Successfully installed d2l-1.0.0a1.post0 jedi-0.18.2 jupyter-1.0.0
qtconsole-5.4.2 qtpy-2.3.1
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Requirement already satisfied: matplotlib_inline in
/usr/local/lib/python3.9/dist-packages (0.1.6)
Requirement already satisfied: traitlets in /usr/local/lib/python3.9/dist-
packages (from matplotlib_inline) (5.7.1)
```

[ ]:
```
%matplotlib inline
import time
import math
import torch
import torchvision
from torchvision import transforms
from d2l import torch as d2l
from torch import nn
import torch.nn.functional as F

d2l.use_svg_display()
```

1.1 Problem 1

For the problem of Machine Translation problem: Train a deeper Transformer than what we did during the lectures. How does it affect the training speed, model complexity, and translation performance both quantitatively and qualitatively? Report and plot your results.

Normalization and other utility functions:

[ ]:
```
class PositionWiseFFN(nn.Module):
    """The positionwise feed-forward network."""
    def __init__(self, ffn_num_hiddens, ffn_num_outputs):
        super().__init__()
        self.dense1 = nn.LazyLinear(ffn_num_hiddens)
        self.relu = nn.ReLU()
        self.dense2 = nn.LazyLinear(ffn_num_outputs)

    def forward(self, X):
        return self.dense2(self.relu(self.dense1(X)))
```

[ ]:
```
ffn = PositionWiseFFN(4, 8)
ffn.eval()
ffn(torch.ones((2, 3, 4)))[0]
```

```
/usr/local/lib/python3.9/dist-packages/torch/nn/modules/lazy.py:180:
UserWarning: Lazy modules are a new feature under heavy development so changes
to the API or functionality can happen at any moment.
  warnings.warn('Lazy modules are a new feature under heavy development '
```

[ ]:
```
tensor([[ 0.2107, -0.2126,  0.1207, -0.3937, -0.1298, -0.3482, -0.1620,
-0.3778],
        [ 0.2107, -0.2126,  0.1207, -0.3937, -0.1298, -0.3482, -0.1620,
-0.3778],
        [ 0.2107, -0.2126,  0.1207, -0.3937, -0.1298, -0.3482, -0.1620,
-0.3778]],
       grad_fn=<SelectBackward0>)
```

[ ]:
```
ln = nn.LayerNorm(2)
bn = nn.LazyBatchNorm1d()
X = torch.tensor([[1, 2], [2, 3]], dtype=torch.float32)
# Compute mean and variance from X in the training mode
print('layer norm:', ln(X), '\nbatch norm:', bn(X))
```

```
layer norm: tensor([[-1.0000,  1.0000],
        [-1.0000,  1.0000]], grad_fn=<NativeLayerNormBackward0>)
batch norm: tensor([[-1.0000, -1.0000],
        [ 1.0000,  1.0000]], grad_fn=<NativeBatchNormBackward0>)
```

[ ]:
```
class AddNorm(nn.Module):
    """The residual connection followed by layer normalization."""
```

```
    def __init__(self, norm_shape, dropout):
        super().__init__()
        self.dropout = nn.Dropout(dropout)
        self.ln = nn.LayerNorm(norm_shape)

    def forward(self, X, Y):
        return self.ln(self.dropout(Y) + X)
```

[ ]:
```
add_norm = AddNorm(4, 0.5)
shape = (2, 3, 4)
d2l.check_shape(add_norm(torch.ones(shape), torch.ones(shape)), shape)
```

The encoder:

[ ]:
```
class TransformerEncoderBlock(nn.Module):
    """The Transformer encoder block."""
    def __init__(self, num_hiddens, ffn_num_hiddens, num_heads, dropout,
                 use_bias=False):
        super().__init__()
        self.attention = d2l.MultiHeadAttention(num_hiddens, num_heads,
                                                dropout, use_bias)
        self.addnorm1 = AddNorm(num_hiddens, dropout)
        self.ffn = PositionWiseFFN(ffn_num_hiddens, num_hiddens)
        self.addnorm2 = AddNorm(num_hiddens, dropout)

    def forward(self, X, valid_lens):
        Y = self.addnorm1(X, self.attention(X, X, X, valid_lens))
        return self.addnorm2(Y, self.ffn(Y))
```

[ ]:
```
X = torch.ones((2, 100, 24))
valid_lens = torch.tensor([3, 2])
encoder_blk = TransformerEncoderBlock(24, 48, 8, 0.5)
encoder_blk.eval()
d2l.check_shape(encoder_blk(X, valid_lens), X.shape)
```

[ ]:
```
class TransformerEncoder(d2l.Encoder):
    """The Transformer encoder."""
    def __init__(self, vocab_size, num_hiddens, ffn_num_hiddens,
                 num_heads, num_blks, dropout, use_bias=False):
        super().__init__()
        self.num_hiddens = num_hiddens
        self.embedding = nn.Embedding(vocab_size, num_hiddens)
        self.pos_encoding = d2l.PositionalEncoding(num_hiddens, dropout)
        self.blks = nn.Sequential()
        for i in range(num_blks):
            self.blks.add_module("block"+str(i), TransformerEncoderBlock(
                num_hiddens, ffn_num_hiddens, num_heads, dropout, use_bias))
```

```
    def forward(self, X, valid_lens):
        # Since positional encoding values are between -1 and 1, the embedding
        # values are multiplied by the square root of the embedding dimension
        # to rescale before they are summed up
        X = self.pos_encoding(self.embedding(X) * math.sqrt(self.num_hiddens))
        self.attention_weights = [None] * len(self.blks)
        for i, blk in enumerate(self.blks):
            X = blk(X, valid_lens)
            self.attention_weights[
                i] = blk.attention.attention.attention_weights
        return X
```

[ ]:
```
encoder = TransformerEncoder(200, 24, 48, 8, 2, 0.5)
d2l.check_shape(encoder(torch.ones((2, 100), dtype=torch.long), valid_lens),
                (2, 100, 24))
```

The decoder:

[ ]:
```
class TransformerDecoderBlock(nn.Module):
    # The i-th block in the Transformer decoder
    def __init__(self, num_hiddens, ffn_num_hiddens, num_heads, dropout, i):
        super().__init__()
        self.i = i
        self.attention1 = d2l.MultiHeadAttention(num_hiddens, num_heads,
                                                 dropout)
        self.addnorm1 = AddNorm(num_hiddens, dropout)
        self.attention2 = d2l.MultiHeadAttention(num_hiddens, num_heads,
                                                 dropout)
        self.addnorm2 = AddNorm(num_hiddens, dropout)
        self.ffn = PositionWiseFFN(ffn_num_hiddens, num_hiddens)
        self.addnorm3 = AddNorm(num_hiddens, dropout)

    def forward(self, X, state):
        enc_outputs, enc_valid_lens = state[0], state[1]
        # During training, all the tokens of any output sequence are processed
        # at the same time, so state[2][self.i] is None as initialized. When
        # decoding any output sequence token by token during prediction,
        # state[2][self.i] contains representations of the decoded output at
        # the i-th block up to the current time step
        if state[2][self.i] is None:
            key_values = X
        else:
            key_values = torch.cat((state[2][self.i], X), dim=1)
        state[2][self.i] = key_values
        if self.training:
            batch_size, num_steps, _ = X.shape
```

```
            # Shape of dec_valid_lens: (batch_size, num_steps), where every
            # row is [1, 2, ..., num_steps]
            dec_valid_lens = torch.arange(
                1, num_steps + 1, device=X.device).repeat(batch_size, 1)
        else:
            dec_valid_lens = None
        # Self-attention
        X2 = self.attention1(X, key_values, key_values, dec_valid_lens)
        Y = self.addnorm1(X, X2)
        # Encoder-decoder attention. Shape of enc_outputs:
        # (batch_size, num_steps, num_hiddens)
        Y2 = self.attention2(Y, enc_outputs, enc_outputs, enc_valid_lens)
        Z = self.addnorm2(Y, Y2)
        return self.addnorm3(Z, self.ffn(Z)), state
```

[ ]:
```
decoder_blk = TransformerDecoderBlock(24, 48, 8, 0.5, 0)
X = torch.ones((2, 100, 24))
state = [encoder_blk(X, valid_lens), valid_lens, [None]]
d2l.check_shape(decoder_blk(X, state)[0], X.shape)
```

[ ]:
```
class TransformerDecoder(d2l.AttentionDecoder):
    def __init__(self, vocab_size, num_hiddens, ffn_num_hiddens, num_heads,
                 num_blks, dropout):
        super().__init__()
        self.num_hiddens = num_hiddens
        self.num_blks = num_blks
        self.embedding = nn.Embedding(vocab_size, num_hiddens)
        self.pos_encoding = d2l.PositionalEncoding(num_hiddens, dropout)
        self.blks = nn.Sequential()
        for i in range(num_blks):
            self.blks.add_module("block"+str(i), TransformerDecoderBlock(
                num_hiddens, ffn_num_hiddens, num_heads, dropout, i))
        self.dense = nn.LazyLinear(vocab_size)

    def init_state(self, enc_outputs, enc_valid_lens):
        return [enc_outputs, enc_valid_lens, [None] * self.num_blks]

    def forward(self, X, state):
        X = self.pos_encoding(self.embedding(X) * math.sqrt(self.num_hiddens))
        self._attention_weights = [[None] * len(self.blks) for _ in range (2)]
        for i, blk in enumerate(self.blks):
            X, state = blk(X, state)
            # Decoder self-attention weights
            self._attention_weights[0][
                i] = blk.attention1.attention.attention_weights
            # Encoder-decoder attention weights
            self._attention_weights[1][
```

```
                i] = blk.attention2.attention.attention_weights
        return self.dense(X), state

    @property
    def attention_weights(self):
        return self._attention_weights
```

Training:

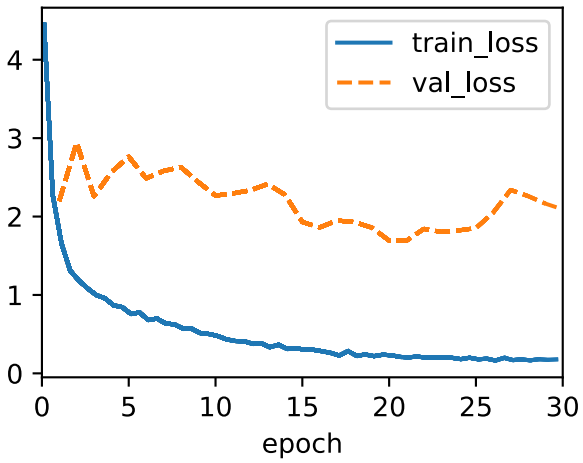
[ ]:
```
data = d2l.MTFraEng(batch_size=128)
```

```
Downloading ../data/fra-eng.zip from
http://d2l-data.s3-accelerate.amazonaws.com/fra-eng.zip…
```

[ ]:
```
num_hiddens, num_blks, dropout = 256, 2, 0.2
ffn_num_hiddens, num_heads = 64, 4

encoder = TransformerEncoder(
    len(data.src_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
decoder = TransformerDecoder(
    len(data.tgt_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
model = d2l.Seq2Seq(encoder, decoder, tgt_pad=data.tgt_vocab['<pad>'],
                    lr=0.0015)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=30, gradient_clip_val=1, num_gpus=1)
trainer.fit(model, data)
```



Test:

[ ]:
```
engs = ['go .', 'i lost .', 'he\'s calm .', 'i\'m home .']
fras = ['va !', 'j\'ai perdu .', 'il est calme .', 'je suis chez moi .']
preds, _ = model.predict_step(
    data.build(engs, fras), d2l.try_gpu(), data.num_steps)
for en, fr, p in zip(engs, fras, preds):
    translation = []
    for token in data.tgt_vocab.to_tokens(p):
        if token == '<eos>':
            break
        translation.append(token)
    print(f'{en} => {translation}, bleu,'
          f'{d2l.bleu(" ".join(translation), fr, k=2):.3f}')
```

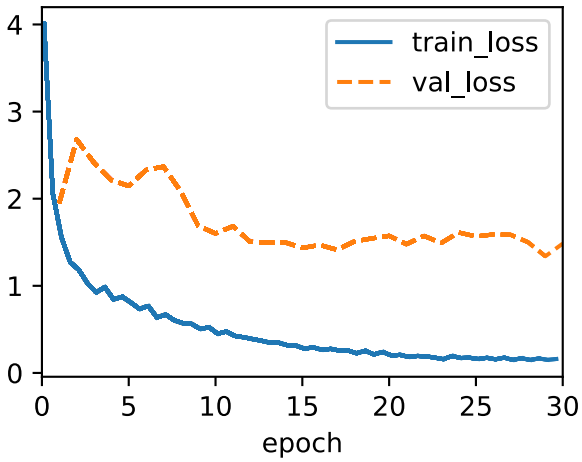
```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['il', 'est', 'mort', '.'], bleu,0.658
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

Make the network deeper (adjust the number of feed-forward layer):

[ ]:
```
num_hiddens, num_blks, dropout = 256, 2, 0.2
ffn_num_hiddens, num_heads = 128, 8

encoder = TransformerEncoder(
    len(data.src_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
decoder = TransformerDecoder(
    len(data.tgt_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
model = d2l.Seq2Seq(encoder, decoder, tgt_pad=data.tgt_vocab['<pad>'],
                    lr=0.0015)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=30, gradient_clip_val=1, num_gpus=1)
trainer.fit(model, data)
```



Test:

[ ]:
```
engs = ['go .', 'i lost .', 'he\'s calm .', 'i\'m home .']
fras = ['va !', 'j\'ai perdu .', 'il est calme .', 'je suis chez moi .']
preds, _ = model.predict_step(
    data.build(engs, fras), d2l.try_gpu(), data.num_steps)
for en, fr, p in zip(engs, fras, preds):
    translation = []
    for token in data.tgt_vocab.to_tokens(p):
        if token == '<eos>':
            break
        translation.append(token)
    print(f'{en} => {translation}, bleu,'
          f'{d2l.bleu(" ".join(translation), fr, k=2):.3f}')
```

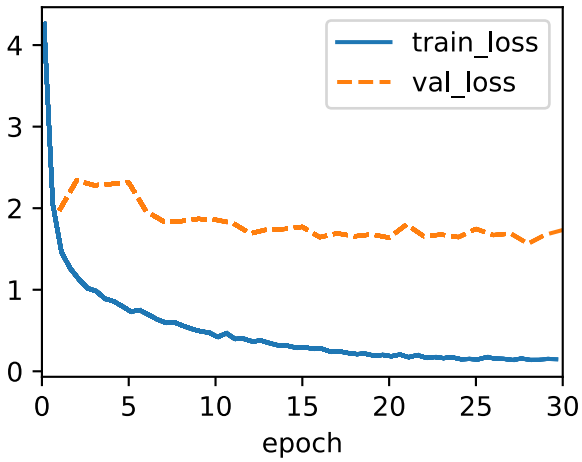
```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['<unk>', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

Adjust the feed-forward layer even more:

[ ]:
```
num_hiddens, num_blks, dropout = 256, 2, 0.2
ffn_num_hiddens, num_heads = 256, 16

encoder = TransformerEncoder(
    len(data.src_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
decoder = TransformerDecoder(
    len(data.tgt_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
model = d2l.Seq2Seq(encoder, decoder, tgt_pad=data.tgt_vocab['<pad>'],
                    lr=0.0015)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=30, gradient_clip_val=1, num_gpus=1)
trainer.fit(model, data)
```



Test:

[ ]:
```
engs = ['go .', 'i lost .', 'he\'s calm .', 'i\'m home .']
fras = ['va !', 'j\'ai perdu .', 'il est calme .', 'je suis chez moi .']
preds, _ = model.predict_step(
    data.build(engs, fras), d2l.try_gpu(), data.num_steps)
for en, fr, p in zip(engs, fras, preds):
    translation = []
    for token in data.tgt_vocab.to_tokens(p):
        if token == '<eos>':
            break
        translation.append(token)
    print(f'{en} => {translation}, bleu,'
          f'{d2l.bleu(" ".join(translation), fr, k=2):.3f}')
```

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['il', 'est', 'mouillé', '.'], bleu,0.658
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

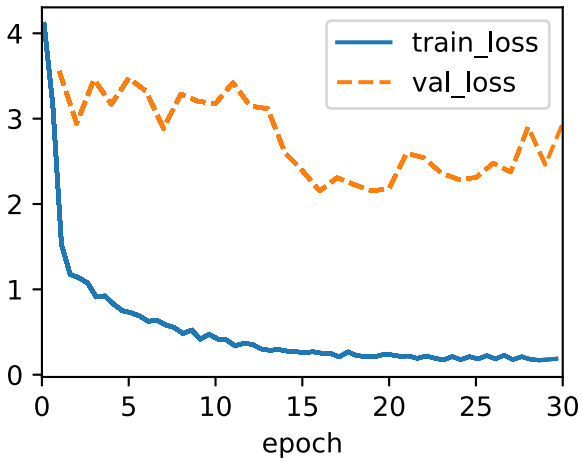
Adjust the number of hidden layers:

[ ]:
```
num_hiddens, num_blks, dropout = 512, 2, 0.2
ffn_num_hiddens, num_heads = 64, 4

encoder = TransformerEncoder(
    len(data.src_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
decoder = TransformerDecoder(
    len(data.tgt_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
model = d2l.Seq2Seq(encoder, decoder, tgt_pad=data.tgt_vocab['<pad>'],
```

```
                      lr=0.0015)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=30, gradient_clip_val=1, num_gpus=1)
trainer.fit(model, data)
```



Test:

[ ]:
```
engs = ['go .', 'i lost .', 'he\'s calm .', 'i\'m home .']
fras = ['va !', 'j\'ai perdu .', 'il est calme .', 'je suis chez moi .']
preds, _ = model.predict_step(
    data.build(engs, fras), d2l.try_gpu(), data.num_steps)
for en, fr, p in zip(engs, fras, preds):
    translation = []
    for token in data.tgt_vocab.to_tokens(p):
        if token == '<eos>':
            break
        translation.append(token)
    print(f'{en} => {translation}, bleu,'
          f'{d2l.bleu(" ".join(translation), fr, k=2):.3f}')
```

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['il', 'est', 'mouillé', '.'], bleu,0.658
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

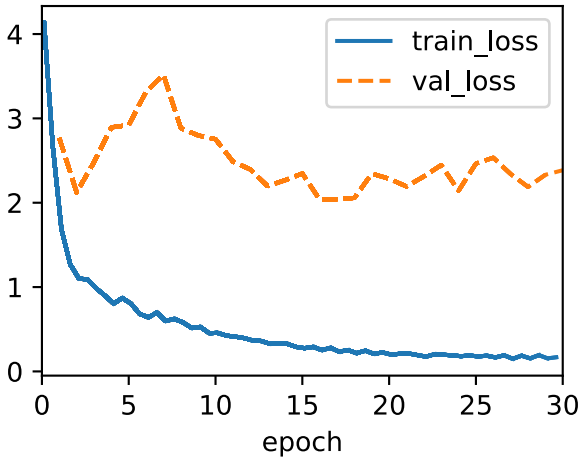
Adjust the number of blocks:

[ ]:
```
num_hiddens, num_blks, dropout = 256, 4, 0.2
ffn_num_hiddens, num_heads = 64, 4

encoder = TransformerEncoder(
    len(data.src_vocab), num_hiddens, ffn_num_hiddens, num_heads,
```

```
    num_blks, dropout)
decoder = TransformerDecoder(
    len(data.tgt_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
model = d2l.Seq2Seq(encoder, decoder, tgt_pad=data.tgt_vocab['<pad>'],
                    lr=0.0015)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=30, gradient_clip_val=1, num_gpus=1)
trainer.fit(model, data)
```



Test:

[ ]:
```
engs = ['go .', 'i lost .', 'he\'s calm .', 'i\'m home .']
fras = ['va !', 'j\'ai perdu .', 'il est calme .', 'je suis chez moi .']
preds, _ = model.predict_step(
    data.build(engs, fras), d2l.try_gpu(), data.num_steps)
for en, fr, p in zip(engs, fras, preds):
    translation = []
    for token in data.tgt_vocab.to_tokens(p):
        if token == '<eos>':
            break
        translation.append(token)
    print(f'{en} => {translation}, bleu,'
          f'{d2l.bleu(" ".join(translation), fr, k=2):.3f}')
```

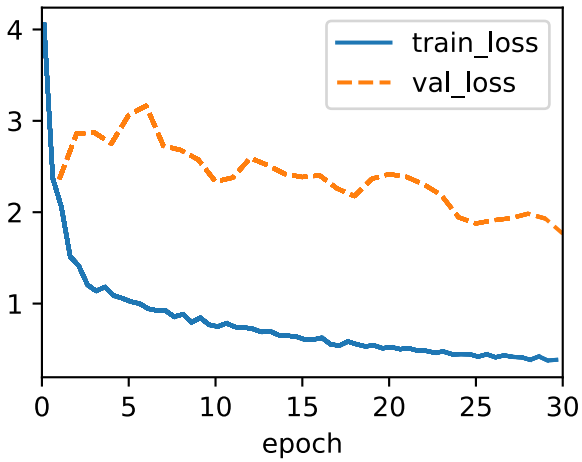
```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['il', 'est', 'mouillé', '.'], bleu,0.658
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

Adjust dropout:

[ ]:
```
num_hiddens, num_blks, dropout = 256, 2, 0.4
ffn_num_hiddens, num_heads = 64, 4

encoder = TransformerEncoder(
    len(data.src_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
decoder = TransformerDecoder(
    len(data.tgt_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
model = d2l.Seq2Seq(encoder, decoder, tgt_pad=data.tgt_vocab['<pad>'],
                    lr=0.0015)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=30, gradient_clip_val=1, num_gpus=1)
trainer.fit(model, data)
```



Test:

[ ]:
```
engs = ['go .', 'i lost .', 'he\'s calm .', 'i\'m home .']
fras = ['va !', 'j\'ai perdu .', 'il est calme .', 'je suis chez moi .']
preds, _ = model.predict_step(
    data.build(engs, fras), d2l.try_gpu(), data.num_steps)
for en, fr, p in zip(engs, fras, preds):
    translation = []
    for token in data.tgt_vocab.to_tokens(p):
        if token == '<eos>':
            break
        translation.append(token)
    print(f'{en} => {translation}, bleu,'
          f'{d2l.bleu(" ".join(translation), fr, k=2):.3f}')
```

```
go . => ['va', 'doucement', '!'], bleu,0.000
```

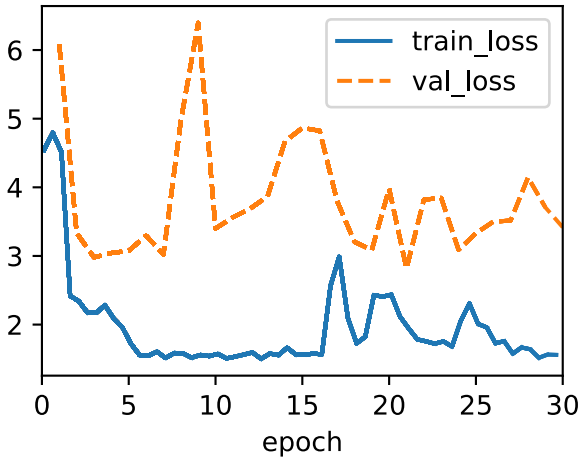
```
i lost . => ['je', 'suis', '<unk>', 'perdu', '.'], bleu,0.447
he's calm . => ['il', 'est', 'calme', '.'], bleu,1.000
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

Combine all together:

[ ]:
```
num_hiddens, num_blks, dropout = 512, 4, 0.4
ffn_num_hiddens, num_heads = 256, 16

encoder = TransformerEncoder(
    len(data.src_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
decoder = TransformerDecoder(
    len(data.tgt_vocab), num_hiddens, ffn_num_hiddens, num_heads,
    num_blks, dropout)
model = d2l.Seq2Seq(encoder, decoder, tgt_pad=data.tgt_vocab['<pad>'],
                    lr=0.0015)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=30, gradient_clip_val=1, num_gpus=1)
trainer.fit(model, data)
```



Test:

[ ]:
```
engs = ['go .', 'i lost .', 'he\'s calm .', 'i\'m home .']
fras = ['va !', 'j\'ai perdu .', 'il est calme .', 'je suis chez moi .']
preds, _ = model.predict_step(
    data.build(engs, fras), d2l.try_gpu(), data.num_steps)
for en, fr, p in zip(engs, fras, preds):
    translation = []
    for token in data.tgt_vocab.to_tokens(p):
        if token == '<eos>':
            break
```

```
        translation.append(token)
    print(f'{en} => {translation}, bleu,'
          f'{d2l.bleu(" ".join(translation), fr, k=2):.3f}')
```

```
go . => [], bleu,0.000
i lost . => [], bleu,0.000
he's calm . => [], bleu,0.000
i'm home . => [], bleu,0.000
```

2 Problem 2

For the problem of the Vision Transformer, we need, in lectures, to train a deeper Transformer with more multiheaded self-attention blocks. How does it affect the training speed, model complexity, and validation accuracy? Report and plot your results.

Embeddings:

[ ]:
```
class PatchEmbedding(nn.Module):
    def __init__(self, img_size=96, patch_size=16, num_hiddens=512):
        super().__init__()
        def _make_tuple(x):
            if not isinstance(x, (list, tuple)):
                return (x, x)
            return x
        img_size, patch_size = _make_tuple(img_size), _make_tuple(patch_size)
        self.num_patches = (img_size[0] // patch_size[0]) * (
            img_size[1] // patch_size[1])
        self.conv = nn.LazyConv2d(num_hiddens, kernel_size=patch_size,
                                  stride=patch_size)

    def forward(self, X):
        # Output shape: (batch size, no. of patches, no. of channels)
        return self.conv(X).flatten(2).transpose(1, 2)
```

[ ]:
```
img_size, patch_size, num_hiddens, batch_size = 96, 16, 512, 4
patch_emb = PatchEmbedding(img_size, patch_size, num_hiddens)
X = torch.zeros(batch_size, 3, img_size, img_size)
d2l.check_shape(patch_emb(X),
                (batch_size, (img_size//patch_size)**2, num_hiddens))
```

The encoder:

[ ]:
```
class ViTMLP(nn.Module):
    def __init__(self, mlp_num_hiddens, mlp_num_outputs, dropout=0.5):
        super().__init__()
        self.dense1 = nn.LazyLinear(mlp_num_hiddens)
        self.gelu = nn.GELU()
        self.dropout1 = nn.Dropout(dropout)
```

```
        self.dense2 = nn.LazyLinear(mlp_num_outputs)
        self.dropout2 = nn.Dropout(dropout)

    def forward(self, x):
        return self.dropout2(self.dense2(self.dropout1(self.gelu(
            self.dense1(x)))))
```

[ ]:
```
class ViTBlock(nn.Module):
    def __init__(self, num_hiddens, norm_shape, mlp_num_hiddens,
                 num_heads, dropout, use_bias=False):
        super().__init__()
        self.ln1 = nn.LayerNorm(norm_shape)
        self.attention = d2l.MultiHeadAttention(num_hiddens, num_heads,
                                                dropout, use_bias)
        self.ln2 = nn.LayerNorm(norm_shape)
        self.mlp = ViTMLP(mlp_num_hiddens, num_hiddens, dropout)

    def forward(self, X, valid_lens=None):
        X = X + self.attention(*([self.ln1(X)] * 3), valid_lens)
        return X + self.mlp(self.ln2(X))
```

[ ]:
```
X = torch.ones((2, 100, 24))
encoder_blk = ViTBlock(24, 24, 48, 8, 0.5)
encoder_blk.eval()
d2l.check_shape(encoder_blk(X), X.shape)
```

Connecting everything:

[ ]:
```
class ViT(d2l.Classifier):
    """Vision Transformer."""
    def __init__(self, img_size, patch_size, num_hiddens, mlp_num_hiddens,
                 num_heads, num_blks, emb_dropout, blk_dropout, lr=0.1,
                 use_bias=False, num_classes=10):
        super().__init__()
        self.save_hyperparameters()
        self.patch_embedding = PatchEmbedding(
            img_size, patch_size, num_hiddens)
        self.cls_token = nn.Parameter(torch.zeros(1, 1, num_hiddens))
        num_steps = self.patch_embedding.num_patches + 1  # Add the cls token
        # Positional embeddings are learnable
        self.pos_embedding = nn.Parameter(
            torch.randn(1, num_steps, num_hiddens))
        self.dropout = nn.Dropout(emb_dropout)
        self.blks = nn.Sequential()
        for i in range(num_blks):
            self.blks.add_module(f"{i}", ViTBlock(
                num_hiddens, num_hiddens, mlp_num_hiddens,
```

```
                num_heads, blk_dropout, use_bias))
        self.head = nn.Sequential(nn.LayerNorm(num_hiddens),
                                  nn.Linear(num_hiddens, num_classes))

    def forward(self, X):
        X = self.patch_embedding(X)
        X = torch.cat((self.cls_token.expand(X.shape[0], -1, -1), X), 1)
        X = self.dropout(X + self.pos_embedding)
        for blk in self.blks:
            X = blk(X)
        return self.head(X[:, 0])
```

Training:

[ ]:
```
data = d2l.FashionMNIST(batch_size=128, resize=(img_size, img_size))
```

```
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-
images-idx3-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-
images-idx3-ubyte.gz to ../data/FashionMNIST/raw/train-images-idx3-ubyte.gz

100%|      | 26421880/26421880 [00:03<00:00, 8666732.59it/s]

Extracting ../data/FashionMNIST/raw/train-images-idx3-ubyte.gz to
../data/FashionMNIST/raw

Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-
labels-idx1-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-
labels-idx1-ubyte.gz to ../data/FashionMNIST/raw/train-labels-idx1-ubyte.gz

100%|      | 29515/29515 [00:00<00:00, 143512.00it/s]

Extracting ../data/FashionMNIST/raw/train-labels-idx1-ubyte.gz to
../data/FashionMNIST/raw

Downloading http://fashion-mnist.s3-website.eu-
central-1.amazonaws.com/t10k-images-idx3-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-
central-1.amazonaws.com/t10k-images-idx3-ubyte.gz to
../data/FashionMNIST/raw/t10k-images-idx3-ubyte.gz

100%|      | 4422102/4422102 [00:01<00:00, 2672794.77it/s]

Extracting ../data/FashionMNIST/raw/t10k-images-idx3-ubyte.gz to
../data/FashionMNIST/raw

Downloading http://fashion-mnist.s3-website.eu-
central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-
```

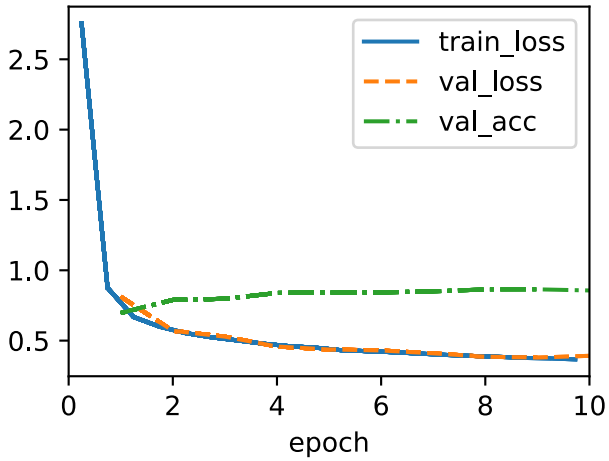
```
central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz to
../data/FashionMNIST/raw/t10k-labels-idx1-ubyte.gz

100%|      | 5148/5148 [00:00<00:00, 5448467.57it/s]

Extracting ../data/FashionMNIST/raw/t10k-labels-idx1-ubyte.gz to
../data/FashionMNIST/raw
```

[ ]:
```
img_size, patch_size = 96, 16
num_hiddens, mlp_num_hiddens, num_heads, num_blks = 512, 2048, 8, 2
emb_dropout, blk_dropout, lr = 0.1, 0.1, 0.1
model = ViT(img_size, patch_size, num_hiddens, mlp_num_hiddens, num_heads,
            num_blks, emb_dropout, blk_dropout, lr)
```

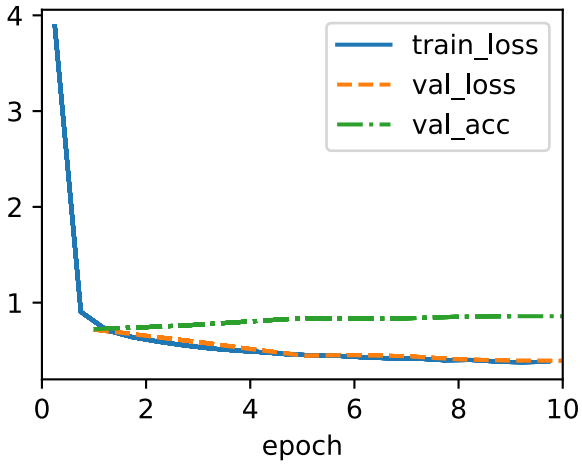
[ ]:
```
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
trainer.fit(model, data)
```



Adjust number of hiddens:

[ ]:
```
img_size, patch_size = 96, 16
num_hiddens, mlp_num_hiddens, num_heads, num_blks = 768, 2048, 8, 2
emb_dropout, blk_dropout, lr = 0.1, 0.1, 0.1
model = ViT(img_size, patch_size, num_hiddens, mlp_num_hiddens, num_heads,
            num_blks, emb_dropout, blk_dropout, lr)
```

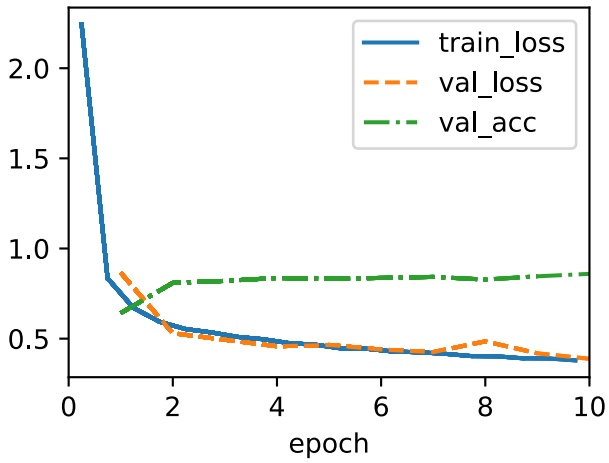
[ ]:
```
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
trainer.fit(model, data)
```



Adjust the number of MLP hiddens:

[ ]:
```
img_size, patch_size = 96, 16
num_hiddens, mlp_num_hiddens, num_heads, num_blks = 512, 4096, 8, 2
emb_dropout, blk_dropout, lr = 0.1, 0.1, 0.1
model = ViT(img_size, patch_size, num_hiddens, mlp_num_hiddens, num_heads,
            num_blks, emb_dropout, blk_dropout, lr)
```

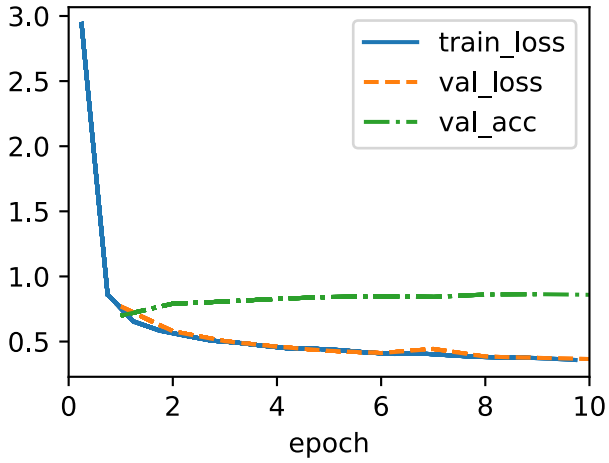
[ ]:
```
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
trainer.fit(model, data)
```



Adjust the number of heads:

[ ]:
```
img_size, patch_size = 96, 16
num_hiddens, mlp_num_hiddens, num_heads, num_blks = 512, 2048, 16, 2
emb_dropout, blk_dropout, lr = 0.1, 0.1, 0.1
model = ViT(img_size, patch_size, num_hiddens, mlp_num_hiddens, num_heads,
            num_blks, emb_dropout, blk_dropout, lr)
```

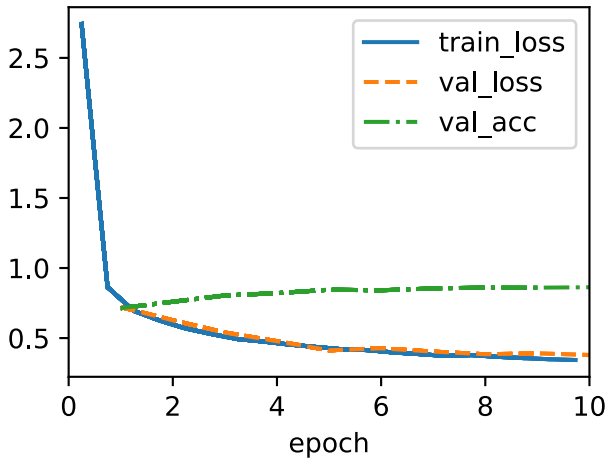
[ ]:
```
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
trainer.fit(model, data)
```



Adjust the number of blocks:

[ ]:
```
img_size, patch_size = 96, 16
num_hiddens, mlp_num_hiddens, num_heads, num_blks = 512, 2048, 8, 4
emb_dropout, blk_dropout, lr = 0.1, 0.1, 0.1
model = ViT(img_size, patch_size, num_hiddens, mlp_num_hiddens, num_heads,
            num_blks, emb_dropout, blk_dropout, lr)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
trainer.fit(model, data)
```



Combine it all together:

[ ]:
```
img_size, patch_size = 96, 16
num_hiddens, mlp_num_hiddens, num_heads, num_blks = 768, 4096, 16, 4
emb_dropout, blk_dropout, lr = 0.1, 0.1, 0.1
model = ViT(img_size, patch_size, num_hiddens, mlp_num_hiddens, num_heads,
            num_blks, emb_dropout, blk_dropout, lr)
```

[ ]:
```
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
trainer.fit(model, data)
```